1. Introduction :

Après avoir donné l'aspect théorique des méthodes de Markov cachés dans le chapitre précédent alors le moment est venu de les appliquer à un problème réel, la poursuite d'une cible radar dans une zone préalablement déterminée.

2. Principe d'utilisation d'un HMM :

Il s'agit de déterminer dans un modèle préalablement appris la séquence d'états correspondant à une observation donnée. On peut utiliser l'algorithme de [Baum Welch] de type "forward-backward" qui calcule la probabilité optimale d'avoir observé une observation O en sommant les probabilités associées à tous les chemins possibles permettant d'obtenir O dans le HMM**.**

Mais l'algorithme le plus utilisé est celui de Viterbi, il ne prend en compte que le meilleur chemin, c'est-à-dire celui qui maximise la probabilité d'obtenir O, l'algorithme de Viterbi est une solution récursive sous-optimale au problème de l'estimation de la séquence d'états dans un HMM**,** cet algorithme est également utilisé dans le cadre plus général de la recherche d'un chemin sous-optimal dans un treillis. [12]

3. Ressources :

3.1 Matérielles :

Pour élaborer notre travail nous avons eu besoin d'un Intel Pentium3 avec la configuration suivante :
Intel Pentium 3 +1.00 GHz
RAM 128 Mo

3.2 Environnement :

Système d'exploitation : Windows XP.

Langage de programmation: Langage de programmation : MATLAB 6.5
Logiciel qui permet de manière interactive :

- de faire des calculs matriciel;
- d'analyser les données;
- de visualiser les résultats en 2D et 3D.

4. Problème posé :

Le problème posé est le suivant: [10]

Nous avons une zone bien déterminées de 6×6 états et 9 capteurs repartis de la manière suivante Voir Figure 4.1. Où chaque état correspond à une position de la cible de coordonnées (m, n), avec$1 \leq m \leq 6,\ 1 \leq n \leq 6$.
Les 9 capteurs sont repartis dans la même zone.

Chaque capteur est en mesure de détecter la cible en mouvement dans la position la plus proche.

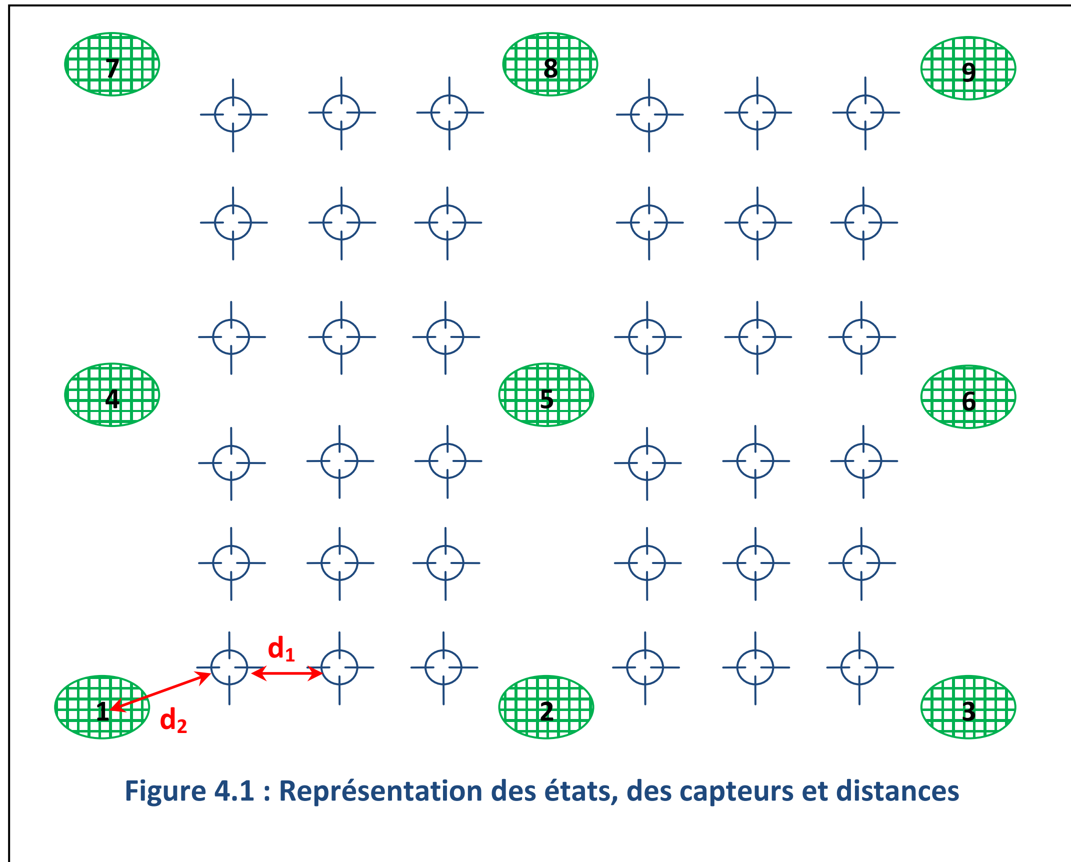
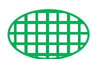


Figure 4.1 : Représentation des états, des capteurs et distances

La légende:

: Représente un capteur dans notre cas c'est un Radar.

 : Représente les états de 1 jusqu'à 36 respectivement de la gauche vers la droite.

d_1 : Représente la distance entre les états (les positions de la cible).

d_2 : Représente la distance entre le capteur et la position de la cible.

Dans notre travail on s'intéresse à faire une expérience qui conserve tous les capteurs ainsi que toutes les positions.

Pour cela on peut traiter trois cas:

4.1 Le premier cas :

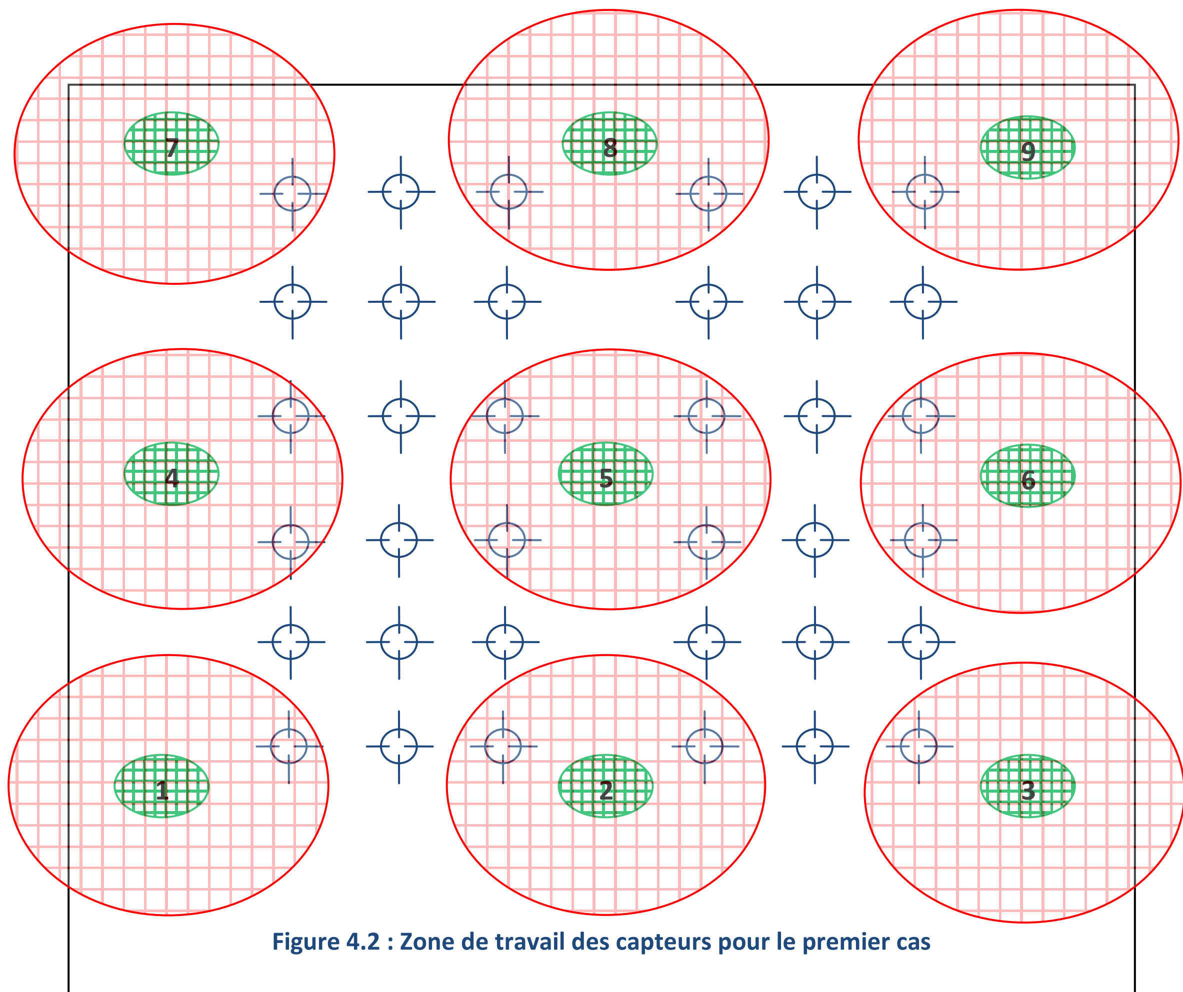


Figure 4.2 : Zone de travail des capteurs pour le premier cas

Capteur 1 : il capte la position 1.
Capteur 2 : il capte les positions 3 et 4.
Capteur 3 : il capte la position 6.
Capteur 4 : il capte les positions 13 et 19.
Capteur 5 : il capte les positions 15 et 16 et 21 et 22.
Capteur 6 : il capte les positions 18 et 24.
Capteur 7 : il capte la position 31.
Capteur 8 : il capte les positions 33 et 34.
Capteur 9 : il capte la position 36.

On remarque que 16 positions sur 36 sont surveillées.

4.2 Le deuxième cas :

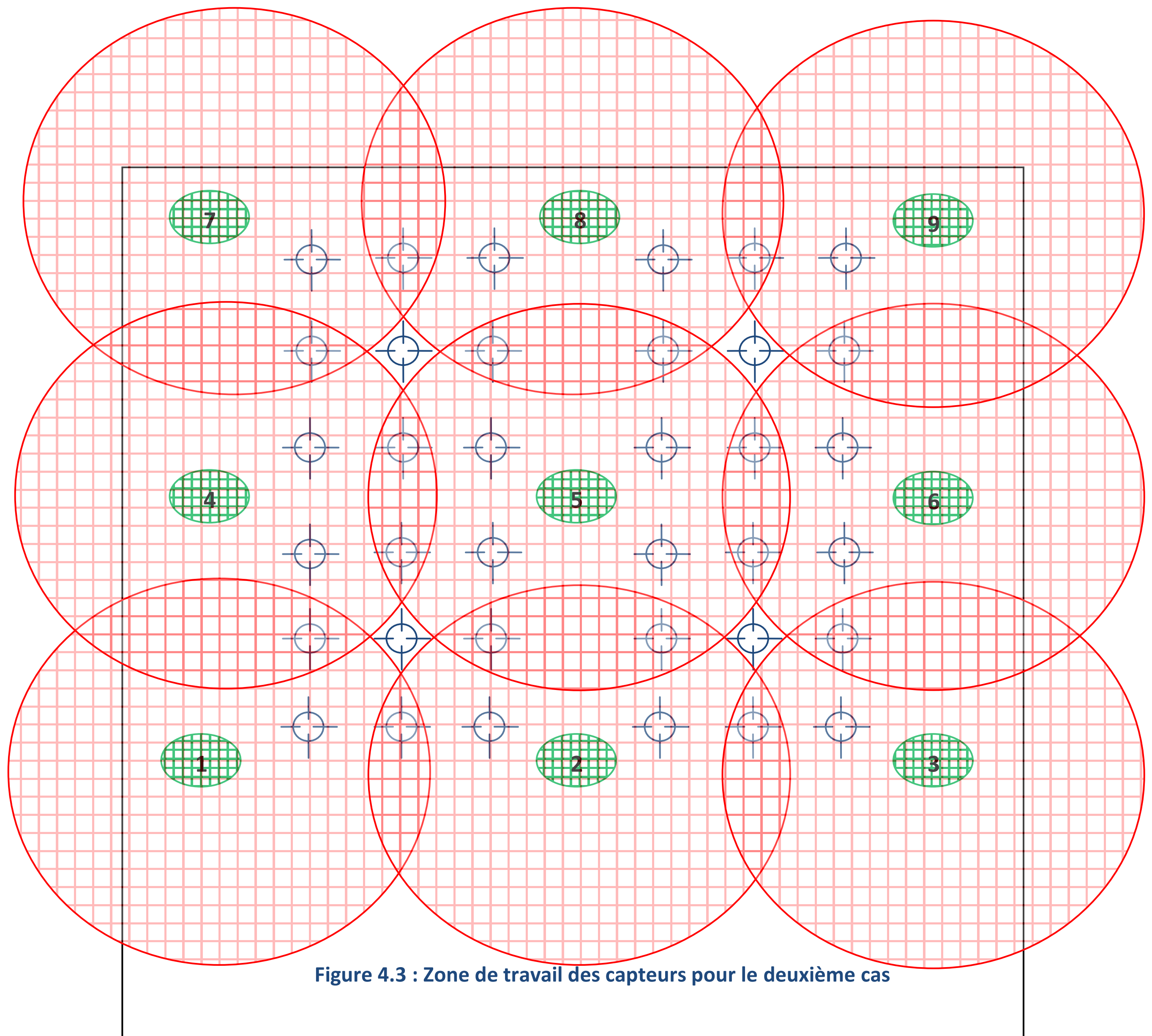


Figure 4.3 : Zone de travail des capteurs pour le deuxième cas

Capteur 1 : il capte les positions 1, 2 et 7.
Capteur 2 : il capte les positions 2, 3, 4, 5, 9 et 10.
Capteur 3 : il capte les positions 5, 6 et 12.
Capteur 4 : il capte les positions 7, 13, 14, 19, 20 et 25.
Capteur 5 : il capte les positions 9, 10, 14, 15, 16, 17, 20, 21, 22, 23, 27 et 28.
Capteur 6 : il capte les positions 12, 17, 18, 23, 24 et 30.
Capteur 7 : il capte les positions 25, 31 et 32.
Capteur 8 : il capte les positions 27, 28, 32, 33, 34 et 35.
Capteur 9 : il capte les positions 30, 35 et 36.

On remarque que 32 positions sur 36 sont surveillées.

4.3 Le troisième cas :

Toutes les positions sont surveillées par les 9 capteurs.

5. Présentation du programme :

5.1 Constitution du corpus :

La base de données est formée d'une matrice de transition (36×36) et d'une matrice d'observation (9×36).

9 : représente le nombre de capteurs
36 : représente le nombre des états (i, j) position dans lesquelles se troue la cible.
La longueur d'observation est fixée à 50

5.2 Les fichiers MATLAB :

Voici la liste du programme et des sous programmes qui seront exécutés sous MATLAB 6.5 sur la plateforme PC Windows:

- HMM-application.m.
- Matrice-transition.m.
- Matrice-observation.m.
- Gene-état-vraie.m.
- Viterbi.m.
- Cumuprod.m
- Calcul-erreur.m.

Et maintenant nous présentons la méthodologie de programme principale.

5.3 Programme principal: HMM-application.m

Ce programme est basé sur les procédures suivantes:

- Localisation et spécification du nombre des états.
- Spécifier la longueur de la séquence du temps pour produire l'observation.
- Appel de la fonction Matrice-transition.m pour générer la matrice de probabilité de transition.
- Appel de la fonction Matrice-observation.m pour déterminer observabilité de chaque capteur.
- Performer 20 itérations. Pour chaque itération, on fait les étapes suivantes:
    - Appel de Gene-état-vraie.m pour générer l'état actuel de la cible.
    - Générer l'observation binaire associée avec chaque observateur, basée sur l'observabilité des capteurs et l'état actuel de la cible.
    - Appel de la fonction Viterbi.m pour trouver l'état estimé.
    - Appel de la fonction Calcul-erreur.m pour calculer et enregistrer les erreurs de distance entre les états actuels et les états estimés.
    - Calcul de l'erreur moyenne entre les positions sur le temps pour l'itération courante.
- Calcul des moyennes est erreurs pour 20 itérations et les représentées dans un graphe.

5.4 Apprentissage:

Le but de l'apprentissage est de déterminer les paramètres qui constituent les modèles tels que la matrice de transition A_{ij} et la matrice d'observation B_{ij}.

- La matrice de transition est déterminée en fonction de la distance qui sépare les états qui est noté d_1.
- La matrice d'observation est déterminée elle aussi en fonction de la distance qui sépare le capteur et la position de la cible qui est notée d_2.

Cas 01 : Le capteur capte la position la plus proche seulement voir Figure 4.2.

Cas 02 : Le capteur capte les deux positions les plus proches voir Figure 4.3.

Cas 03 : Le capteur peut capter jusqu'à trois positions.

6. Résultats, courbes et commentaires :

Cas 01: pour le premier cas on a eu les résultats suivants:

Etat actuel:

13 13 20 32 26 27 35 34 34 34 28 28 28 27 33 33 26 20 31 25 25
25 27 27 34 33 33 28 34 29 35 36 29 34 28 27 32 26 20 22 17 22
9 16 23 29 24 24 22 22

Etat estimé:

25 25 25 25 26 32 32 32 32 32 27 27 27 32 32 32 31 31 31 31 31
31 31 31 31 31 33 28 35 35 35 35 35 35 35 35 36 36 36 36 29 29
29 29 29 29 29 29 29 29

: Les positions entourées d'un cercle rouge ont été identifié (état actuelle = l'état estimé).

La courbe des trajectoires actuel et estimé est représentée dans la figure 4.4.

La courbe d'erreur quadratique moyenne est représentée dans la figure 4.5.

Remarque :

La représentation des courbes d'erreur quadratique moyenne dans les trois cas est basée sur le calcul de l'erreur moyenne sur 20 trajectoires distinctes.

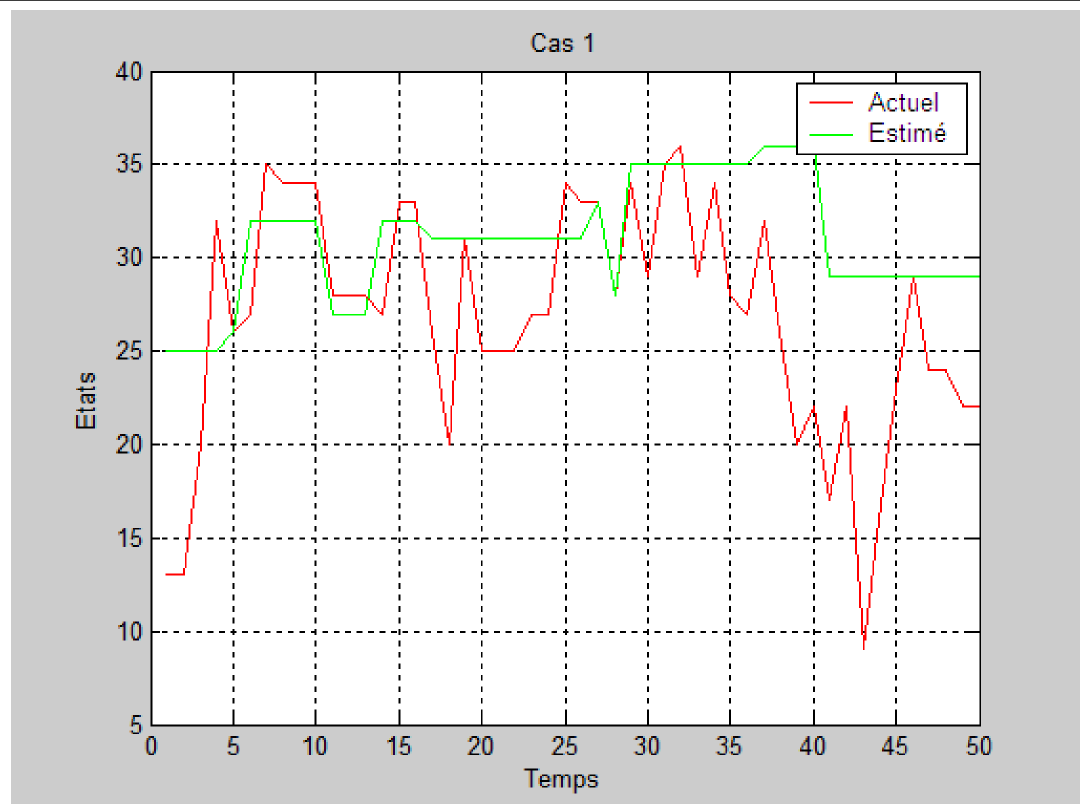


Figure 4.4 : Courbe des trajectoires pour le premier cas

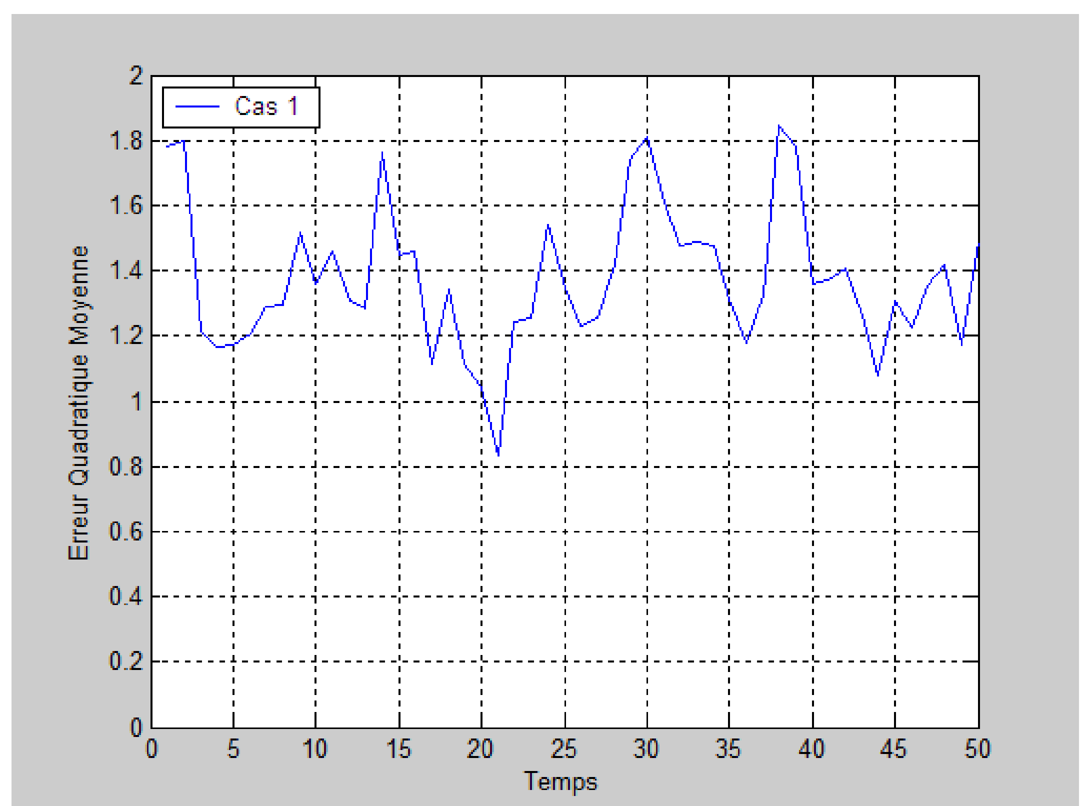


Figure 4.5 : Courbe d’erreur pour le premier cas

Pour le premier cas (cas1), on constate que l'erreur moyenne entre l'état désirée et l'état estimée est assez importante par ce que chaque capteur est capable de capté la position la plus proche seulement.

Cas 2 :

Dans ce cas, nous avons ajusté les paramètres du modèle qui sont en fonction des distances d_1 et d_2. On garde la même matrice de transition du cas 1, et nous avons changé la matrice d'observation qui est en fonction de la distance d_2. C'est-à-dire on a varié d_2 de telle sorte que le capteur capte les deux positions les plus proches ce qui donne 32 position surveiller du totale 36 (voir la figure 4.3).

Etat actuel:

4 3 10 10 22 33 28 28 27 28 30 30 22 20 22 27 35 35 28 34 34

35 35 29 23 24 29 30 30 36 35 36 36 36 36 35 35 35 24 24 23 17

22 33 20 20 9 3 4 4

Etat estimé:

4 4 4 4 4 4 16 28 28 28 34 34 34 34 34 34 34 34 34 34 34

36 36 36 36 36 36 36 36 36 36 36 36 36 36 36 35 35 23 23 23 23

28 22 22 22 22 22 10 10

La courbe des trajectoires actuel et estimé est représentée dans la figure 4.6.

La courbe d'erreur quadratique moyenne est représentée dans la figure 4.7.

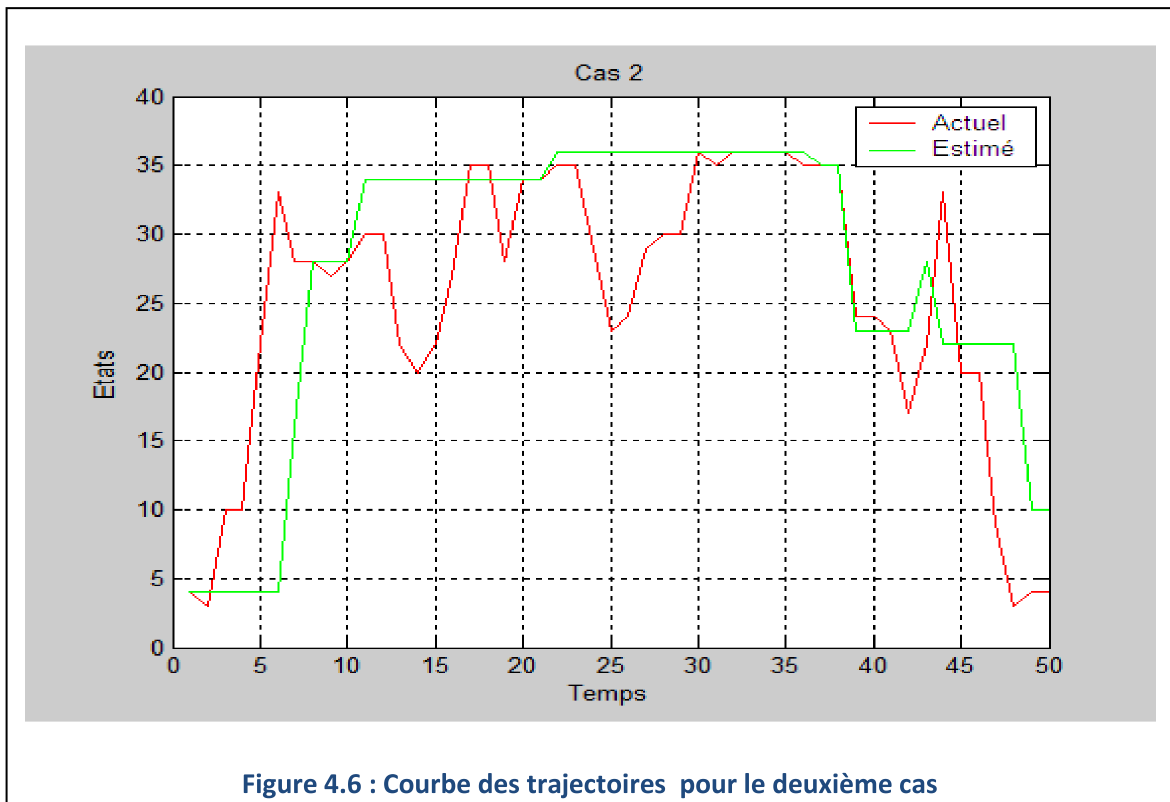


Figure 4.6 : Courbe des trajectoires pour le deuxième cas

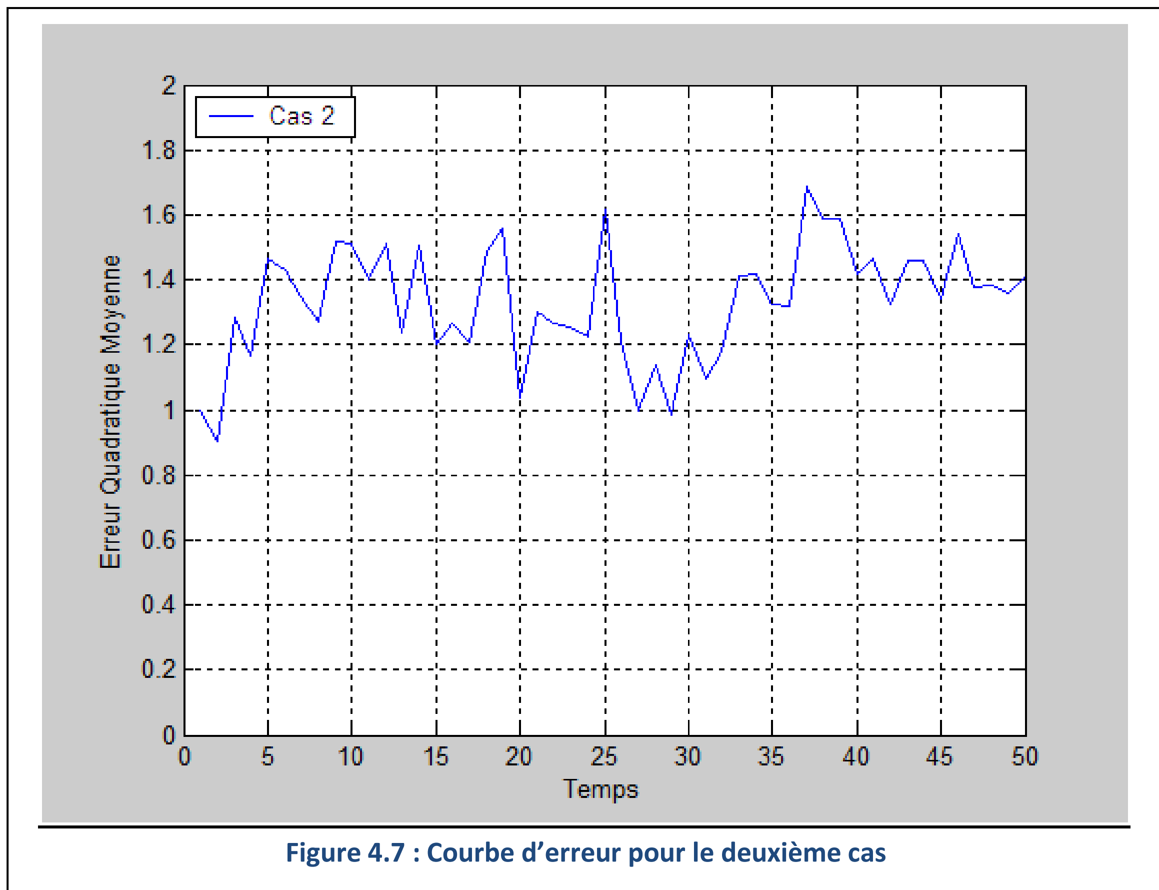


Figure 4.7 : Courbe d'erreur pour le deuxième cas

Comme remarque nous avons constaté une légère amélioration où l'erreur moyenne entre l'état désiré et l'état estimé est importante.

Cas 3

Etat actuel:

1 1 1 1 1 1 1 1 1 1 2 2 1 1 2 2 2 2 3 3 2

2 2 8 9 15 15 15 14 14 14 14 14 14 13 19 19 13 13 13 13 13

13 19 13 19 13 13 7 8

Etat estimé:

1 1 1 1 1 1 1 1 1 1 2 2 2 2 3 3 3 3 3 3 3

3 3 9 9 9 9 9 8 13 13 13 13 13 13 13 13 13 13 13 13 13

13 13 13 13 13 13 14 14

La courbe des trajectoires actuel et estimé est représentée dans la figure 4.8.

La courbe d'erreur quadratique moyenne est représentée dans la figure 4.9.

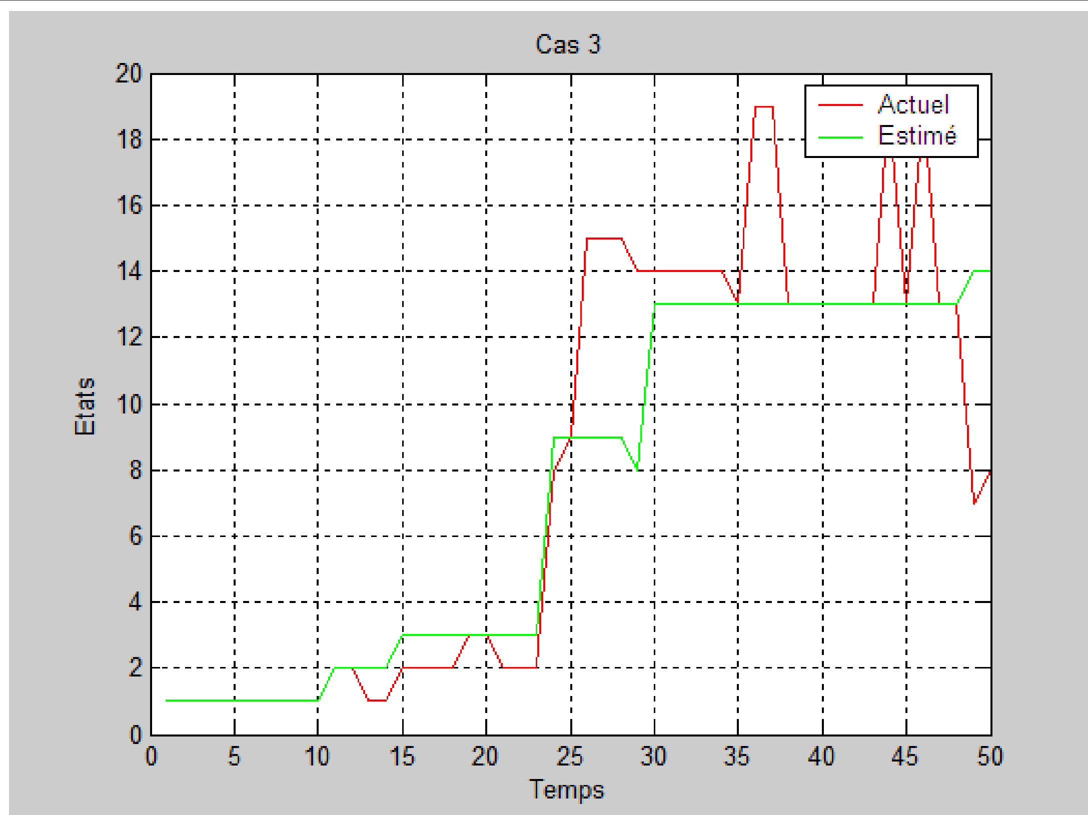


Figure 4.8 : Courbe des trajectoires pour le troisième cas

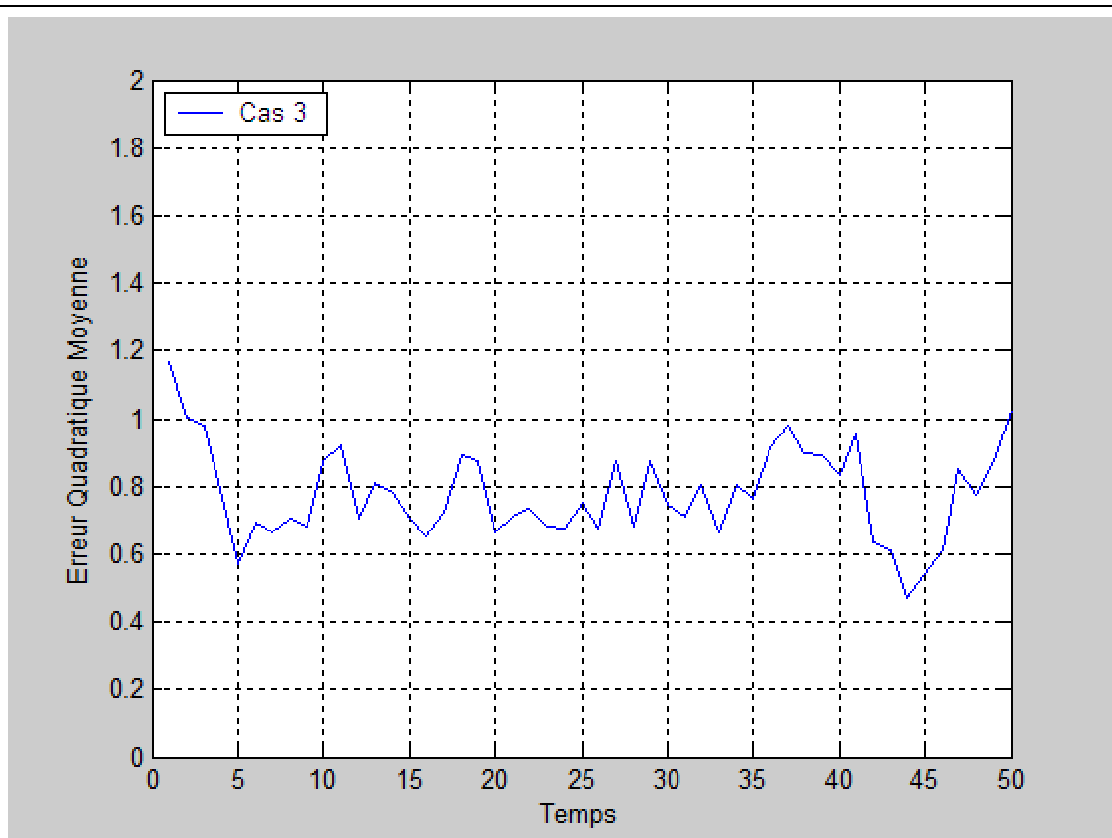


Figure 4.9 : Courbe d'erreur pour le troisième cas

Dans ce cas nous avons réajusté les paramètres du modèle. On a changé la matrice de transition du cas 1 et cas 2 qui est en fonction de la distance d_1, et nous avons gardé la matrice d'observation du cas 2 qui est en fonction de la distance d_2. C'est-à-dire on a varié d_1 de telle sorte que le capteur capte les trois positions les plus proche ce qui donne 36 positions surveiller et on obtenue des résultats satisfaisants (on constat une nette amélioration).

La figure suivante représente l'erreur des trois cas en même temps.

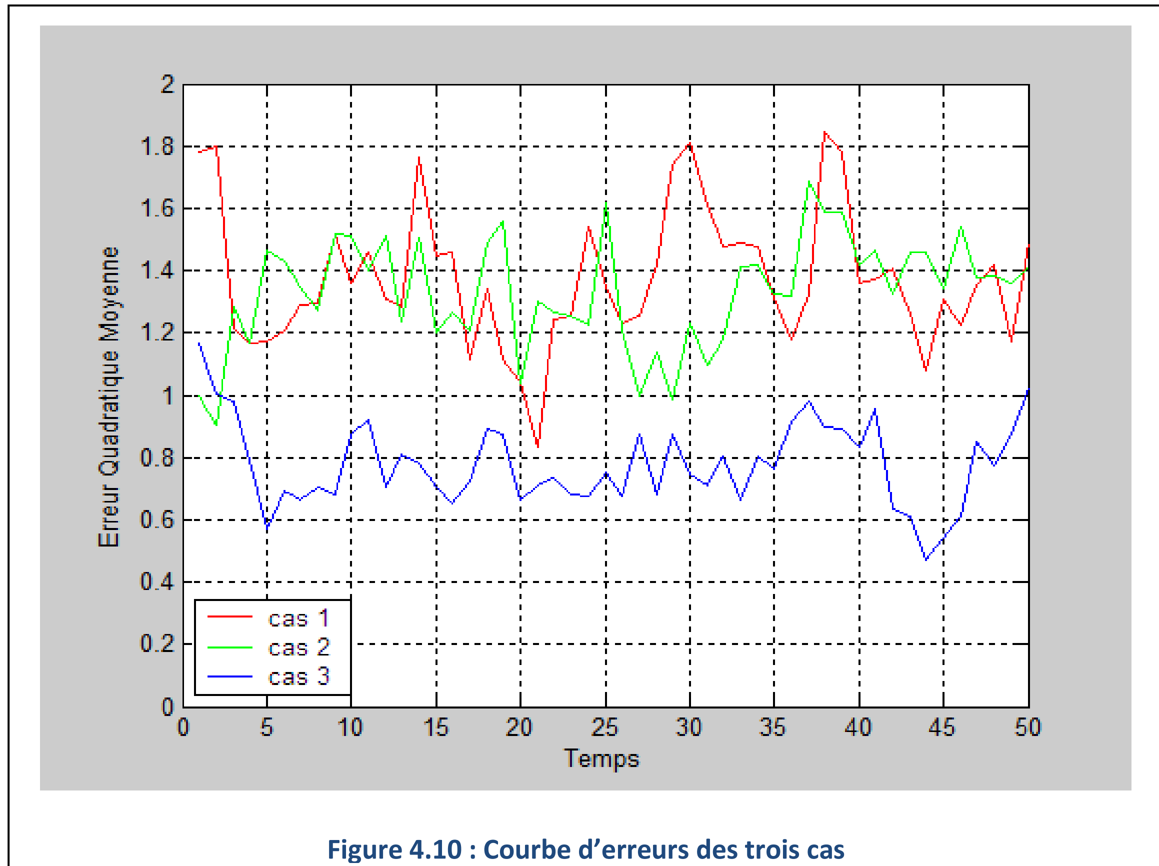


Figure 4.10 : Courbe d'erreurs des trois cas

La figure 4.10 montre que le troisième cas est le meilleur par rapport aux deux autre cas.

Donc on peut conclure que les meilleurs résultats peuvent être obtenus par la diminution de la distance d_1.

7. Conclusion :

Dans ce chapitre nous avons simulé une application « La poursuite d'une cible RADAR dans une zone préalablement déterminée » par la technique des modèles de Markov cachés (HMM).
La variation de la distance entre les états d'une part et entre les capteurs et les états d'autres part ; par l'utilisation de l'algorithme de Viterbi qui nous a permis des résultats satisfaisants par cette technique (figure 4.10).
On peut dire que c'est un résultat encourageant pour ce genre de problème, et que la technique (HMM) restera toujours valable dans plusieurs domaines d'application malgré leur ancienne apparition.